Proceedings of ALGORITMY 2012
pp. 331–341

A COMPARISON OF SEVERAL TIME DISCRETIZATION METHODS FOR FREE SURFACE FLOWS *

EBERHARD BÄNSCH AND STEPHAN WELLER†

Abstract. The fundamental problems in the numerical approximation of multiphase systems, or more generally speaking, in the treatment of flows with capillary free boundaries are the representation of the free surface, evaluating the curvature, handling of discontinuities, most importantly the pressure jump, and time discretization strategies for the decoupling of flow computation and geometry.

While for the first three items there exists a vast literature and many techniques developed over the last two decades, the last problem of how to efficiently treat the time discretization has been widely ignored. The majority of the existing approaches just decouple the flow field from the geometry by a simple segregated approach, i.e. evaluating the geometric quantities from the previous time step. These strategy leads to a) a severe *capillary* CFL condition and b) is of first order in time at most. Existing semi-implicit discretizations exist that overcome problem a), but are still first order only and rather dissipative in certain situations.

We compare different existing time discretization schemes for the fully coupled system of Navier-Stokes and the phase boundary condition. Time discretization might be semi-implicit, fully implicit or linearly implicit. The methods are applied to a simple test case where experimental order of convergence and numerical dissipation can be compared relatively easily.

Key words. Free surface flow, Multiphase flow, Higher order time discretization, Finite elements

AMS subject classifications. 76T99

1. Introduction.

Compared to one-phase flows, multiphase systems exhibit several fundamental problems. Among them are:

- the representation of the free surface;
- the evaluation of the curvature;
- handling of discontinuities, most importantly the pressure jump, thus trying to minimize *spurious velocities*;
- time discretization, especially strategies for the decoupling of flow field computation and geometry.

Many techniques have been developed to analyze the first three problems, time discretization has been examined far less rigorously. The most common decoupling approach is a simple segregated approach. The geometric quantities (e.g. curvature) are taken from the previous time step and the flow computation is carried out as if no free boundary was present. The computational domain is then updated and so on. While being relatively easy to implement, this approach has several disadvantages:

- This explicit decoupling scheme is only conditionally stable, more precisely, a severe CFL of the kind

$$\Delta t \leq C\sqrt{We}\, h^{\frac{3}{2}}$$

  is observed numerically. A linearized stability analysis [13] is in agreement with this finding. Though this CFL can be be improved to $\Delta t \leq Ch$, where C does *not* depend on surface tension via regularization techniques [13], it cannot be overcome completely.

*This work has been supported by Deutsche Forschungsgemeinschaft, Priority Programme 1506 *Transport Processes at Fluidic Interfaces*.

† both: Applied Mathematics III, Dept. Mathematics, Univ. Erlangen, Cauerstr. 11, 91058 Erlangen, Germany

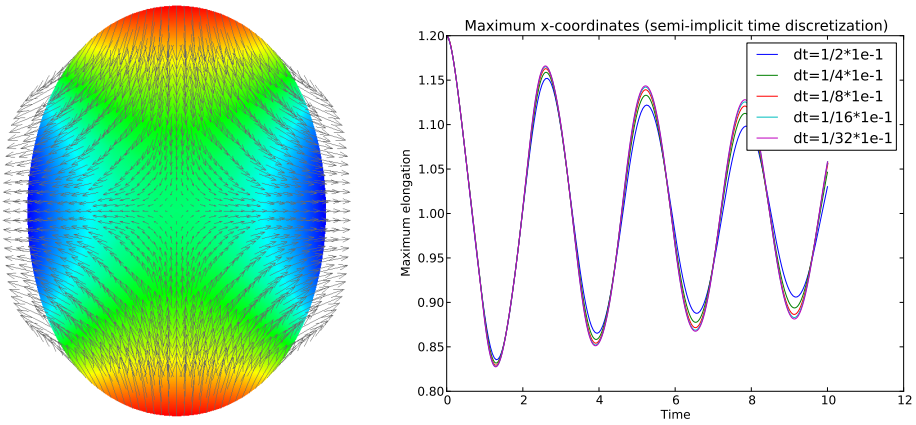

FIG. 1.1. *Simulation of a liquid drop, from [15], and illustration of strong numerical dissipativity of a semi-implicit time discretization scheme.*

- The scheme is of first order only. This is also inherent to the explicit decoupling and cannot be overcome with such a simple decoupling.

In [3], a semi-implicit decoupling was used to achieve an unconditionally stable scheme. While solving the stability problems completely, this scheme is still of first order only. Furthermore, it suffers from high numerical dissipation at large time step sizes.

If the maximum elongation of an oscillating drop (i.e. we consider one-phase flow with a free boundary) is plotted against time at various time step sizes, this dissipation becomes quite prominent (see Fig. 1.1).

There is no systematic analysis of any discretization technique in time available for the coupled problem. Thus, how to best discretize in time the fully coupled system of Navier-Stokes equations and surface conditions is completely open.

2. Mathematical model. We consider the nondimensionalized Navier-Stokes equations on a time dependent, a priori unknown domain $\Omega = \Omega(t) \subset \mathbb{R}^d$, $d \in \{2,3\}$, with a jump condition on a phase boundary Γ:

$$\begin{aligned} \partial_t \mathbf{u} + \mathbf{u}\cdot\nabla\mathbf{u} + \nabla\mathbf{p} - \frac{1}{\mathrm{Re}}\Delta\mathbf{u} &= \rho f, \\ \nabla\cdot\mathbf{u} &= 0, \\ [\![S\nu]\!]_\Gamma &= \sigma\kappa\nu. \end{aligned}$$

Here, $S = S(\mathbf{u},\mathbf{p}) = \frac{1}{\mathrm{Re}} D[\mathbf{u}] - \mathbf{p}I = \frac{1}{\mathrm{Re}}(\nabla\mathbf{u} + (\nabla\mathbf{u})^\top) - \mathbf{p}I$ is the stress tensor, σ is the surface tension coefficient and ν is the unit normal vector on the phase boundary Γ. $[\![\cdot]\!]_\Gamma$ describes the jump over the phase boundary. Existence of solutions for short time or small data have been known for a long time, see e.g. [12, 14]

For the jump condition across the phase boundary we use a variational formulation of the curvature, so a "natural" weak formulation can be derived:

$$e(\Gamma,\varphi) = \int_\Gamma \underline{\nabla}\mathrm{id}\cdot\underline{\nabla}\varphi,$$

where Γ is the phase boundary, φ a test function and $\underline{\nabla}$ is the tangential gradient. This

formulation goes back to Dziuk [6]. Choosing suitable bilinear forms for the Navier-Stokes equations we arrive at a coupled system of the following form:

$$\begin{aligned} m_{\Omega(t)}(\partial_t \mathbf{u}, \mathbf{v}) + n_{\Omega(t)}(\mathbf{u}; \mathbf{v}, \mathbf{u}) + k_{\Omega(t)}(\mathbf{u}, \mathbf{v}) - b_{\Omega(t)}(p, \mathbf{v}) + e(\Gamma, \mathbf{v}) &= m_{\Omega(t)}(f, \mathbf{v}), \\ b_{\Omega(t)}(q, \mathbf{u}) &= 0, \\ V_\Gamma \cdot \nu &= \mathbf{u} \cdot \nu, \end{aligned} \tag{2.1}$$

where V_Γ is the velocity of the phase boundary Γ, and the bilinear and trilinear forms are defined as follows:

$$\begin{aligned} m_\Omega(\mathbf{u}, \mathbf{v}) &:= \int_\Omega \rho \mathbf{u} \cdot \mathbf{v} && \mathbf{u}, \mathbf{v} \in \mathscr{U}, \\ k_\Omega(\mathbf{u}, \mathbf{v}) &:= \frac{1}{2} \int_\Omega \eta\, D[\mathbf{u}] : D[\mathbf{v}] && \mathbf{u}, \mathbf{v} \in \mathscr{U}, \\ n_\Omega(\mathbf{u}; \mathbf{v}, \mathbf{w}) &:= \int_\Omega \rho w \cdot (\nabla \mathbf{u}) \mathbf{v} && \mathbf{u}, \mathbf{v}, w \in \mathscr{U}, \\ b_\Omega(q, \mathbf{v}) &:= \int_\Omega q \nabla \cdot \mathbf{v} && q \in \mathscr{L}^2(\Omega), \mathbf{v} \in \mathscr{U}, \\ e(\Gamma, \mathbf{v}) &:= \int_\Gamma \underline{\nabla} \mathrm{id} : \underline{\nabla} \mathbf{v} && \mathbf{v} \in \mathscr{U}, \end{aligned} \tag{2.2}$$

where $\mathscr{U} \subset H^1(\Omega)$ is a suitable function space.

Since the space $\mathscr{U}$ and all bilinear forms depend on Ω and therefore on time, no straightforward application of a time discretization scheme is possible.

Written on a reference domain $\hat{\Omega}$ with a mapping $\Phi(t, \cdot) : \hat{\Omega} \to \Omega(t)$, and utilizing arbitrary Lagrange-Eulerian (ALE) coordinates, we can transform those equations to:

$$\begin{aligned} m_\Phi(\partial_t \hat{\mathbf{u}}, \hat{\mathbf{v}}) + n_\Phi(\hat{\mathbf{u}}; \hat{\mathbf{v}}, \hat{\mathbf{u}}) + k_\Phi(\hat{\mathbf{u}}, \hat{\mathbf{v}}) - b_\Phi(\hat{p}, \hat{\mathbf{v}}) + e_\Phi(\Gamma, \hat{\mathbf{v}}) &= m_\Phi(\hat{f}, \hat{\mathbf{v}}), \\ b_\Phi(\hat{q}, \hat{\mathbf{u}}) &= 0, \\ V_\Gamma \cdot \nu &= \mathbf{u} \cdot \nu, \end{aligned} \tag{2.3}$$

for all $\hat{\mathbf{v}} \in \mathscr{U}$, where $\hat{\mathbf{u}} \in \mathscr{U}, \hat{p} \in \mathscr{L}^2(\hat{\Omega})$, $\mathscr{U} \subset H^1(\hat{\Omega})^d$ a suitable subspace, and the bilinear forms are accordingly transformed, more precisely:

$$\begin{aligned} m_\Phi(\hat{\mathbf{u}}, \hat{\mathbf{v}}) &:= \int_{\hat{\Omega}} \hat{\rho} \hat{\mathbf{u}} \cdot \hat{\mathbf{v}} |\det D\Phi| && \hat{\mathbf{u}}, \hat{\mathbf{v}} \in \mathscr{U}, \\ k_\Phi(\hat{\mathbf{u}}, \hat{\mathbf{v}}) &:= \frac{1}{2} \int_{\hat{\Omega}} \hat{\eta}\, \hat{D}[\hat{\mathbf{u}}] : \hat{D}[\hat{\mathbf{v}}] |\det D\Phi| && \hat{\mathbf{u}}, \hat{\mathbf{v}} \in \mathscr{U}, \\ n_\Phi(\hat{\mathbf{u}}; \hat{\mathbf{v}}, \hat{\mathbf{w}}) &:= \int_{\hat{\Omega}} \hat{\rho} (\hat{\mathbf{w}} - \dot{\Phi}) \cdot (\nabla \hat{\mathbf{u}}) \hat{\mathbf{v}} |\det D\Phi| && \hat{\mathbf{u}}, \hat{\mathbf{v}}, \hat{\mathbf{w}} \in \mathscr{U}, \\ b_\Phi(\hat{q}, \hat{\mathbf{v}}) &:= \int_{\hat{\Omega}} \hat{q} (D\Phi)^{-1} : \nabla \hat{\mathbf{v}} |\det D\Phi| && \hat{q} \in \mathscr{L}^2(\hat{\Omega}), \hat{\mathbf{v}} \in \mathscr{U}, \\ e_\Phi(\mathbf{v}) &:= \int_{\Phi(\hat{\Gamma})} \underline{\nabla} \mathrm{id} : \underline{\nabla} \mathbf{v} && \hat{\mathbf{v}} \in \mathscr{U}, \end{aligned} \tag{2.4}$$

where $\hat{D}[\hat{\mathbf{v}}] = D\Phi^{-\top} \hat{\nabla} \hat{\mathbf{v}} + \hat{\nabla} \hat{\mathbf{v}}^\top D\Phi^{-1}$.

Note that the dependency of the bilinear forms on the domain can now explicitly be seen in the form of the $D\Phi$-terms. Since we transformed our bilinear forms, the domain is no longer time-dependent, but *the bilinear forms are*.

3. Space discretization. Choosing a standard finite element approach, let $\phi_i, i = 1,\dots,n_v$ form a basis of some finite-dimensional subspace of $H^1(\Omega)^d$, $\psi_i, i = 1,\dots,n_p$ a basis of some finite-dimensional subspace of $\mathscr{L}^2(\Omega)$. Furthermore, let $U = (U_i)_i, P = (P_i)_i$ denote the vectors of coefficients of some function u, p with respect to this basis functions and define the matrices:

$$\begin{aligned} M[\Phi]_{ij} &= m_\Phi(\varphi_j, \varphi_i), \\ N[U,\Phi]_{ij} &= n_\Phi(\sum_k U_k \varphi_k, \varphi_j, \varphi_i), \\ K[\Phi]_{ij} &= k_\Phi(\varphi_j, \varphi_i), \\ B^t[\Phi]_{ij} &= b_\Phi(\psi_j, \varphi_i). \end{aligned}$$

Since all bilinear forms depend on the (time-dependent) domain and hence on the boundary, let us introduce a finite-dimensional basis used as a parametrization of the boundary. Locally on a triangle, the identity on the boundary piece Γ_l can be represented as $\mathrm{id}_{\Gamma_l} = \sum_{k=1}^{n_b} X_k \eta_k$. We can then represent the boundary integral on this triangle as

$$e_{\Gamma_l}(\varphi_i) = \int_{\Gamma_l} \underline{\nabla} \sum_k X_k \eta_k : \underline{\nabla} \varphi_i. \tag{3.1}$$

Define the matrix

$$E[\Phi]_{ij} = \int_{\Phi(\hat{\Gamma})} \underline{\nabla} \eta_j : \underline{\nabla} \varphi_i. \tag{3.2}$$

From now on, we will realize the equation $V_\Gamma \cdot \nu = \mathbf{u} \cdot \nu$ by choosing $V_\Gamma = \mathbf{u}$. Another possible choice is $V_\Gamma = \mathbf{u} \cdot \nu \nu$, however, this leads to more technical difficulties (as the normals are of course dependent on Γ).

Now, we can define the restriction operator to the boundary as

$$R_{ij} = \begin{cases} 1 & \text{if } \eta_j = \varphi_{i|\Gamma} \\ 0 & \text{otherwise,} \end{cases}$$

and we can write down the space discrete system:

$$\begin{aligned} M[\Phi]\dot{U} + N[U,\dot{\Phi}]U + K[\Phi]U + B[\Phi]P + E[\Phi]X &= F[\Phi], \\ B[\Phi]^\top U &= 0, \\ -RU + \dot{X} &= 0. \end{aligned} \tag{3.3}$$

Note that this system lacks the description of the inner part of Ω, i.e. Φ is only defined uniquely at the boundary. In an ALE framework, however, the interior part of Φ is only important for practical aspects (non-degenerate grid etc.) and can be chosen freely.

4. Time discretization. We will now offer three different approaches to discretize System (3.3) in time. Because of the nonlinearity in the transformation Φ, it is clear that a straightforward time discretization will lead to a highly nonlinear system. If this is not wanted, there are various approaches that lead to linear systems. The semi-implicit approach described in the next subsection has been developed in [1], whereas the fully implicit approach and the linearly implicit approach are described in [4].

4.1. Semi-implicit time discretization. The semi-implicit time discretization approach used in [1] is mainly based on the fact that the third equation in (3.3), when discretized by an implicit Euler scheme, can be solved directly. Assuming constant time step size τ and using the usual convention $U^n := U(t^n)$, an implicit Euler scheme yields

$$X^{n+1} = X^n + \tau R U^{n+1}. \tag{4.1}$$

Plugging this in the curvature term $E[\Phi]X$ (assuming an implicit treatment of this term) yields:

$$\begin{aligned} E[\Phi^{n+1}]X^{n+1} &= E[\Phi^{n+1}](X^n + \tau R U^{n+1}) \\ &\approx E[\Phi^n]X^n + \tau E[\Phi^n] R U^{n+1}. \end{aligned} \tag{4.2}$$

The approximation in the second line obviously is an explicit treatment, but the treatment of X^{n+1} is implicit. It turns out that the implicit approximation of X^{n+1} is crucial for stability, even if all other terms are treated explicitly in Φ.

Since the approximation in (4.2) is explicit in Φ but implicit in U, it decouples the Navier-Stokes equations from the equation describing the domain movement. This allows for a very effective solution of the whole system.

The time derivative $\dot{\Phi}$ in the convection part of the system is also treated explicitly by an extrapolation formula.

In principle, any time discretization scheme can now be used to solve the Navier-Stokes part of the equations. In [1], a fractional step θ-splitting which allows for an effective treatment of the solenoidal condition, was used. Of course the order of convergence of the whole method is limited by the semi-implicit treatment in equation (4.2) and can never be higher than 1. Also, there is no easy way to extend this treatment by using a higher oder scheme for this equation, since the explicit treatment of Φ still is a first-order approximation [15].

4.2. Fully implicit time discretization by BDF2. For a fully implicit time discretization approach, any ODE time discretization method may be used. As a suitable candidate, we choose the *Backward Differentiation Formula (BDF)* of second order. As a solver for stiff ordinary differential equations, backward differentiation formulas have been used since the 1970s. Since the second order BDF2 is strongly A-stable and has optimal convergence properties with respect to the second Dahlquist barrier [5], it is widely used. In the context of projection methods [7], BDF2 has been used for the Navier-Stokes equation, however, no application that includes a free surface boundary term is known to us (except [4]).

A straightforward application of BDF2 to (3.3) yields:

$$\begin{aligned} M[\Phi^{n+1}]\left(\frac{3U^{n+1} - 4U^n + U^{n-1}}{2\tau}\right) + N[U^{n+1}, \tilde{\Phi}]U^{n+1} + \\ K[\Phi^{n+1}]U^{n+1} + B[\Phi^{n+1}]P^{n+1} + E[\Phi^{n+1}]X^{n+1} = F[\Phi^{n+1}], \\ B[\Phi^{n+1}]^\top U = 0, \\ -RU^{n+1} + \left(\frac{3X^{n+1} - 4X^n + X^{n-1}}{2\tau}\right) = 0, \end{aligned} \tag{4.3}$$

where $\tilde{\dot{\Phi}}$ is an approximation to $\dot{\Phi}$ using the same BDF2 treatment for the parametrization X of Φ as in the last line.

Note that Solving this system is difficult due to the nonlinearity in Φ, possible approaches will be described in Section 5.

4.3. Linearly implicit time discretization using Runge-Kutta methods. While explicit Runge-Kutta methods have stability properties that make them unsuitable for strongly stiff systems, implicit Runge-Kutta methods are better suited for such problems. Strongly A-stable implicit Runge-Kutta methods of high order are readily available. However, the solution of implicit Runge-Kutta method requires the inversion of a nonlinear system, which is too expensive in many applications. Thus, methods were developed based on the idea to apply one step of a Newton iteration to an implicit Runge-Kutta system. The resulting schemes are called linearly implicit Runge-Kutta methods (LIRK).

LIRK schemes require only the solution of a fixed number of linear systems per time step, while retaining many of the properties, especially wrt. stability, of implicit Runge-Kutta methods. However, the execution of one Newton step requires the Jacobian or at least a good approximation of it has to be available. Methods based on the use of the exact Jacobian are called Rosenbrock methods, while methods utilizing an approximation are called W-methods.

Both Rosenbrock- and W-methods have been investigated in the context of parabolic differential equations [11, 8]. Theoretical results show that for W-methods, second-order convergence can be achieved if the approximation of the Jacobian matrix is good enough (under suitable smoothness assumption [11]).

For the Navier-Stokes equations, Rosenbrock-methods have been used successfully [9]. The Jacobian is relatively easy to compute in this case and is quite close to the matrix generated by an Oseen problem. We were able to verify numerically that the method shows third order convergence in this case.

To apply LIRK methods to a free surface flow example, one has to get a good approximation of the Jacobian, i.e. to the term

$$J_{\cdot j} = \begin{pmatrix} \partial_{U_j} M[X]^{-1}(A[U,X]U + B[X]P + E[X]X) \\ \partial_{P_j} M[X]^{-1}(A[U,X]U + B[X]P + E[X]X) \\ \partial_{X_j} M[X]^{-1}(A[U,X]U + B[X]P + E[X]X) \end{pmatrix}. \tag{4.4}$$

The Jacobian with respect to U and P, i.e. the first two rows of the matrix, are easily computable (exactly as the Jacobian for Navier-Stokes on a fixed domain). The last line, however, is harder to compute. The main problem is the term $\partial_X M[X]^{-1}$. This can be seen by applying the chain rule. The term $\partial_X M[X]^{-1} = M[X]^{-1}(\partial_X M[X])M[X]^{-1}$ can be computed, but it is not a sparse matrix (even if neglecting the first part $M[X]^{-1}$, which can be treated by multiplying the fully discrete-equation by $M[X]$). It is therefore not possible to use this Jacobian matrix, at least not in a standard finite element discretization.

Instead of using the Jacobian, we suggest two different approximations:

1. Complete neglect of all nonlinearities (0th order approximation):

$$J = \begin{pmatrix} M[X]^{-1}A[U,X] & M[X]^{-1}B[X] & M[X]^{-1}E[X]) \\ M[X]^{-1}B[X]^{\top} & 0 & 0 \\ R[X] & 0 & I \end{pmatrix}. \tag{4.5}$$

   This is identical to the system matrix of the original Navier-Stokes system using a semi-implicit time discretization, i.e. we can apply an existing solver in this case.

2. Neglect of the nonlinearities in the mass matrix (1st order approximation):

$$J_{\cdot j} = \begin{pmatrix} M[X]^{-1}\partial_{U_j}(A[U,X]U + B[X]P + E[X]X) \\ M[X]^{-1}\partial_{P_j}(A[U,X]U + B[X]P + E[X]X) \\ M[X]^{-1}\partial_{X_j}(A[U,X]U + B[X]P + E[X]X) \end{pmatrix} \tag{4.6}$$

   The remaining terms can be computed explicitly in an ALE-FEM setting, however, an finite difference approximation might be easier to implement in other settings.

5. Solution techniques for the fully nonlinear discrete system. The fully discrete system (4.3) is highly nonlinear, since almost all terms depend on the computational domain at time t^n.

For the sake of brevity, let us introduce the following abbreviations:

$$\chi^n := \begin{pmatrix} U^n \\ P^n \\ X^n \end{pmatrix}; \Psi^n := \begin{pmatrix} A[U^n,\Gamma^n] & \frac{2\Delta t}{3}B[\Gamma^n] & \frac{2\Delta t}{3}E[\Gamma^n] \\ B[\Gamma^n]^\top & & \\ -\frac{2\Delta t}{3}R[\Gamma^n] & & I \end{pmatrix}; \eta := \begin{pmatrix} G \\ 0 \\ \frac{4}{3}X^n - \frac{1}{3}X^{n-1} \end{pmatrix}.$$

where $A[U,\Phi]$ and G subsume all corresponding terms from (4.3).

Equation (4.3) can now be rewritten in the following form:

$$\chi^{n+1} = \Upsilon^{-1}\eta - \Upsilon^{-1}\Psi^{n+1}\chi^{n+1} + \chi^{n+1}, \tag{5.1}$$

where Υ is an arbitrary (invertible) preconditioner. The resulting fixed point iteration is

$$\chi^{n+1(k)} = (I - \Upsilon^{-1}\Psi^{n+1})\chi^{n+1(k-1)} + \Upsilon^{-1}\eta. \tag{5.2}$$

The convergence rate is determined by the spectral radius of the iteration matrix $I - \Upsilon^{-1}\Psi^{n+1}$. If Υ is close to Ψ^{n+1}, the spectral radius will be small and yield fast convergence.

5.1. First order semi-implicit time discretization schemes as a preconditioner. As a special case of the fixed point iteration in the last section, we can choose $\Upsilon = \Psi^{n-1}$. This would be equivalent to choosing the explicit time discretization of the same problem as a preconditioner. To achieve better stability properties, it is also possible to choose a semi-implicit time discretization $\Upsilon = \tilde{\Psi}^n$. More precisely, consider the time discretization scheme

$$\begin{pmatrix} \tilde{A}[\tilde{U},\tilde{\Phi}] & \frac{2\Delta t}{3}B[\tilde{\Phi}] & \frac{2\Delta t}{3}\tilde{E}[\tilde{\Phi}] \\ B[\tilde{\Phi}]^\top & & \\ -\frac{2\Delta t}{3}R & & I \end{pmatrix} \begin{pmatrix} U^{n+1} \\ P^{n+1} \\ X^{n+1} \end{pmatrix} = \begin{pmatrix} G \\ 0 \\ \frac{4}{3}X^n - \frac{1}{3}X^{n-1} \end{pmatrix}, \tag{5.3}$$

where $\tilde{U},\tilde{\Phi}$ are extrapolations of U^n,Φ^n. For a semi-implicit approach [3], we have

$$X^{n+1} = \frac{4}{3}X^n - \frac{1}{3}X^{n-1} + \frac{2\Delta t}{3}RU^{n+1} \tag{5.4}$$

and we can choose:

$$\tilde{A}[U,\Phi] = A[U,\Phi] + (\frac{2\Delta t}{3})^2 E[\Phi]R. \tag{5.5}$$

With $\tilde{U} = U^{n+1}, \tilde{\Phi} = \Phi^n$ we now get a semi-implicit approach where velocity and boundary treatment are decoupled. While this approach alone is of first order only, it can be used as a preconditioner Υ to solve the fully implicit system given above.

6. Numerical results.

6.1. Example setups. As an example to measure convergence properties and also numerical dissipation, an oscillating 2d drop with a capillary boundary (cf. Fig. 1.1) is used. The drop is perturbed from its stationary state (a perfect circle) by 20%. Initial velocity is chosen as 0. As the capillary forces drive the drop towards its steady state, it starts oscillating. Due to viscous forces, the oscillation is dampened and the tip of the drop moves as a dampened oscillation. This oscillation can be approximately described by the function

$$\exp(-\mu t)(\lambda_1\cos(t) + \lambda_2\sin(t)), \tag{6.1}$$

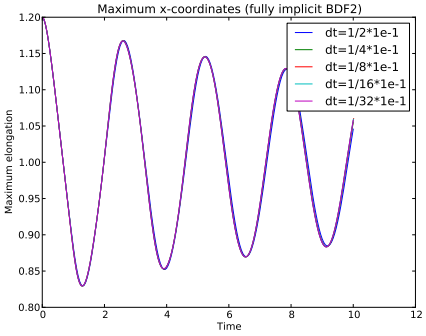

FIG. 6.1. *Maximum elongation of an oscillating drop, showing the significantly lower dissipativity of BDF2 (compare Fig. 1.1)*

which is the solution to $\ddot{x} + \mu\dot{x} + \lambda x = 0$. A parameter fit is used to match this equation to the movement of the tip of the oscillating drop, and the resulting parameter μ then describes the damping of the oscillation of the drop. For a perfect time discretization, this should be independent of time step size (no *numerical* dissipation). We use this parameter to measure how much additional dissipation is introduced by the time discretization. For all runs, a Weber number of 1 and a Reynolds number of 100 were used.

For all examples, the experimental order of convergence was measured and the numerical dissipation was compared to the semi-implicit scheme.

The finite element framework of the existing solver NAVIER [2] was used for implementation.

6.2. Numerical dissipation properties. The numerical dissipation coefficient μ from equation (6.1) was compared for the semi-implicit time discretization, the fully implicit time discretization, and the linearly implicit ROS3P method. In Fig. 6.3, we plot the error in *mu*, i.e. the difference of *mu* for various time step sizes compared to *mu* for a very small time step size (the value of μ for $\Delta t \to 0$ does not differ significantly for the compared methods).

Plots of the elongation illustrating the lower dissipativity are shown in Fig. 6.1 and 6.2, respectively.

6.3. Convergence properties of implicit BDF2 time discretization. In this example a linearization of the original equations (cf. Section 5) has been used as a preconditioner to solve the implicit BDF2 time discretization. The iteration count of the fixed point iteration was 3 to 7 iterations in this example.

The experimental order of convergence measured shows clearly a second order convergence, and also the numerical dissipation is one order of magnitude lower than in the semi-implicit scheme. The numerical error of this and all other methods in this paper is shown in Fig. 6.4

6.4. Convergence properties of linearly implicit time discretization using ROS3P. As an example of a linearly implicit time discretization method, the ROS3P method [10] was used. It is a three-stage Rosenbrock method that is of third order (however theoretical observations [11] show that with an approximate Jacobian only second order convergence can be achieved). It is a strongly A-stable diagonal Runge-Kutta method, i.e. in every time step only one Jacobian matrix (or approximation thereof) has to be assembled. Also, the

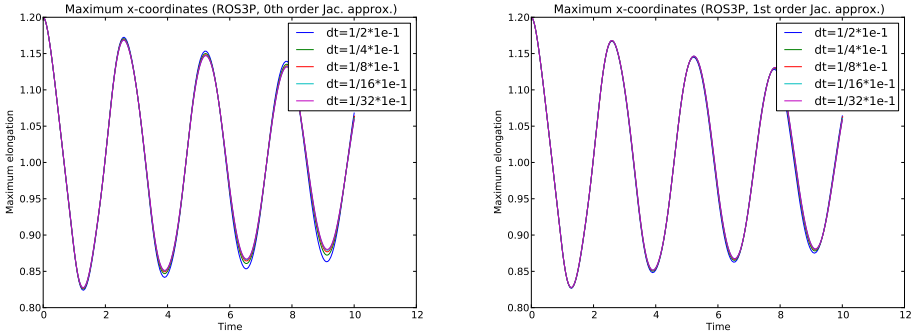

FIG. 6.2. *Maximum elongation of an oscillating drop, showing the significantly lower dissipativity of ROS3P with both Jacobian approximation described in this paper (compare Fig. 1.1)*

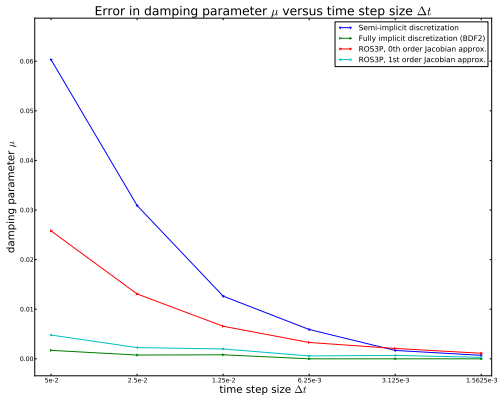

FIG. 6.3. *Damping parameter μ versus time step size Δt for the ROS3P method. Fit against the ODE $\ddot{x}+\mu\dot{x}+\lambda x=0$.*

evaluation of the right hand side is required only twice. For easier comparison, the error of both method has been plotted together with all other methods in Fig. 6.4.

6.5. Convergence properties of the semi-implicit time discretization. For comparison, the classical semi-implicit method [2] was also tested for convergence properties. As the semi-implicit ansatz for the curvature is based on an implicit Euler discretization, no more than first order convergence can be expected. The numerical results are in good agreement with this fact (Fig. 6.4).

7. Conclusion. We compared two new approaches for time discretization of free surface flows, implicit and linearly implicit methods with an existing semi-implicit scheme.

The fully implicit scheme displays the best convergence properties; second order convergence is numerically evident. However, this scheme is also the most expensive scheme. For practical use, this scheme is only competitive, if a good preconditioner is used. Obviously good approximations to the Jacobian matrix are a good candidate here (so then the fixed point iteration used to solve the system becomes an approximate Newton's method). Unfortunately the exact Jacobian remains intractable as of yet.

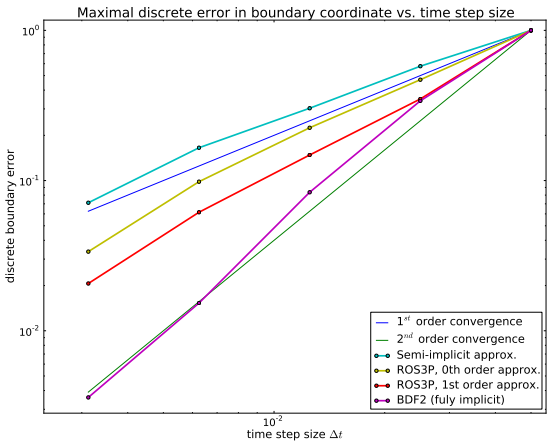

FIG. 6.4. *Error of the ROS3P method (0th order and 1st order Jacobian approximation), the fully implicit BDF2 scheme, and a semi-implicit time discretization. Normalized to initial error of* 1 *for better comparability.*

A good approximation of the Jacobian is also the main point for the linearly implicit methods. A coarse approximation shows results that are only marginally better than the semi-implicit approximation (at comparable numerical costs). None of the approximations considered in this work can be proven to be close to the original Jacobian in terms of the time step size Δt. However, such an approximation would be necessary to achieve second order convergence [11].

REFERENCES

[1] Bänsch, E.: Numerical methods for the instationary Navier–Stokes equations with a free capillary surface. Habilitationsschrift, Universität Freiburg (1998)
[2] Bänsch, E.: Simulation of instationary, incompressible flows. Acta mathematica Universitatis Comenianae **67, no. 1**, 101–114 (1998)
[3] Bänsch, E.: Finite element discretization of the Navier-Stokes equations with a free capillary surface. Numer. Math. **88**, 203–235 (2001)
[4] Bänsch, E., Weller, S.: Fully implicit time discretization for a free surface flow problem. PAMM **11**(1), 619–620 (2011). DOI 10.1002/pamm.201110299. URL http://dx.doi.org/10.1002/pamm.201110299
[5] Deuflhard, P., Bornemann, F.: Numerische Mathematik II. De Gruyter (2002)
[6] Dziuk, G.: An algorithm for evolutionary surfaces. Num. Math. **58**, 603–611 (1991)
[7] Guermond, J.L., Minev, P., Shen, J.: An overview of projection methods for incompressible flows. Computer Methods in Applied Mechanics and Engineering **195, Issues 44-47**, 6011–6045 (2005)
[8] Lang, J.: Adaptive Multilevel Solution of Nonlinear Parabolic PDE Systems. Theory, Algorithm, and Applications. In: Lecture Notes in Computational Sciences and Engineering, vol. 16. Springer (2000)
[9] Lang, J., Teleaga, I.: Higher-order linearly implicit one-step methods for three-dimensional incompressible navier-stokes equations. Studia Univ. "Babeş-Bolyai", Mathematica **LIII**(1) (2008)
[10] Lang, J., Verwer, J.: ROS3P – An accurate third-order Rosenbrock solver designed for parabolic prbolems . Bit Numerical Mathematics (2001)
[11] Lubich, C., Ostermann, A.: Linearly implicit time discretization of non-linear parabolic equations. IMA Journal Num. Anal. **15(4)**, 555–583 (1995)
[12] Solonnikov, V.A.: The solvability of the problem concerning the evolution of an isolated volume of viscous incompressible capillary fluid. J. Soviet Math. **32**, 223–228 (1986)
[13] Sussman, M., Ohta, M.: A stable and efficient method for treating surface tension in incompressible two-phase flow. SIAM J. Sci. Comp. **31(4)** (2009)
[14] Wagner, A.: Nonstationary Marangoni convection. Appl. Math. (Warsaw) **26**(2), 195–220 (1999)

[15] Weller, S.: Higher Order Time Discretization for Free Surface Flows. Diploma thesis, University of Erlangen (2008)